



별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto  
is a true copy from the records of the Korean Intellectual  
Property Office.

출 원 번 호 : 10-2003-0058891  
Application Number

출 원 년 월 일 : 2003년 08월 25일  
Date of Application AUG 25, 2003

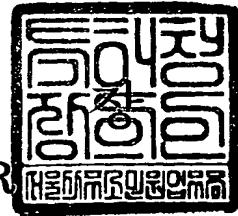
출 원 인 : 삼성전자주식회사  
Applicant(s) SAMSUNG ELECTRONICS CO., LTD.



2003 년 10 월 15 일

특 허 청

COMMISSIONER



## 【서지사항】

【서류명】	분할출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0021
【제출일자】	2003.08.25
【국제특허분류】	G06F
【발명의 명칭】	마크업 문서의 버퍼링 상태를 제어하기 위한 제어 정보가 기록된 정보저장매체, 그 재생 장치 및 재생 방법
【발명의 영문명칭】	Information storage medium containing control information for controlling buffering status of markup document, reproducing method and apparatus therefor
【출원인】	
【명칭】	삼성전자 주식회사
【출원인코드】	1-1998-104271-3
【대리인】	
【성명】	이영필
【대리인코드】	9-1998-000334-6
【포괄위임등록번호】	2003-003435-0
【대리인】	
【성명】	이해영
【대리인코드】	9-1999-000227-4
【포괄위임등록번호】	2003-003436-7
【발명자】	
【성명의 국문표기】	정현권
【성명의 영문표기】	CHUNG, Hyun Kwon
【주민등록번호】	721217-1042731
【우편번호】	135-120
【주소】	서울특별시 강남구 신사동 569번지 302호
【국적】	KR
【발명자】	
【성명의 국문표기】	허정권
【성명의 영문표기】	HEO, Jung Kwon
【주민등록번호】	681207-1830616

【우편번호】	137-766		
【주소】	서울특별시 서초구 반포2동 주공아파트 2단지 203동 504호		
【국적】	KR		
【발명자】			
【성명의 국문표기】	고정완		
【성명의 영문표기】	K0, Jung Wan		
【주민등록번호】	600925-1119917		
【우편번호】	442-707		
【주소】	경기도 수원시 팔달구 망포동 벽산아파트 114동 1101호		
【국적】	KR		
【원출원의표시】			
【출원번호】	10-2003-0058695		
【출원일자】	2003.08.25		
【심사청구일자】	2003.08.25		
【우선권주장】			
【출원국명】	KR		
【출원종류】	특허		
【출원번호】	10-2002-0063631		
【출원일자】	2002.10.17		
【증명서류】	첨부		
【우선권주장】			
【출원국명】	KR		
【출원종류】	특허		
【출원번호】	10-2003-0027073		
【출원일자】	2003.04.29		
【증명서류】	첨부		
【심사청구】	청구		
【취지】	특허법 제52조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인 이영필 (인) 대리인 이해영 (인)		
【수수료】			
【기본출원료】	20	면	29,000 원
【가산출원료】	35	면	35,000 원

1020030058891

출력 일자: 2003/10/20

【우선권주장료】	2 건	43,000 원
【심사청구료】	10 항	429,000 원
【합계】	536,000 원	
【첨부서류】	1. 요약서·명세서(도면)_1통 2. 우선권증명서류 및 동 번역문_2통	

### 【요약서】

#### 【요약】

본 발명은 마크업 문서의 버퍼 상태를 제어하기 위한 정보가 기록된 정보저장매체, 그 재생 장치 및 재생 방법에 관한 것이다.

본 발명에 따른 AV 데이터를 마크업 문서를 사용하여 인터랙티브 모드로 재생하는 재생 장치는 상기 마크업 문서를 버퍼링하는 버퍼; 상기 버퍼를 관리하여 상기 마크업 문서를 상기 버퍼에 프리로드하는 버퍼 매니저; 및 상기 버퍼에 버퍼링되어 있는 마크업 문서를 가져와서 해석하고 디코딩하는 컨텐트 디코더를 포함하고, 상기 버퍼 매니저는 상기 버퍼의 프리로드 완료 여부를 상기 컨텐트 디코더로 알려주는 것을 특징으로 한다. 이에 따라, DVD에 기록된 AV 데이터를 재생하여 마크업 문서를 통해 디스플레이하는 경우 동영상 화면과 연관된 마크업 문서를 디스플레이하고자 할 때, 마크업 문서의 버퍼링 상태를 조사할 수 있으므로, 디스크의 물리적 파손 또는 네트워크의 오동작(연결 끊김)에 따라 마크업 문서를 버퍼링할 수 없게 되었을 때, 즉, 재생과 다운로드 지연 현상과 같은 오동작 발생시에도 대체 마크업 문서를 활용하거나 자연스러운 넘기기 동작이 가능하게 된다.

#### 【대표도】

도 14

**【명세서】****【발명의 명칭】**

마크업 문서의 버퍼링 상태를 제어하기 위한 제어 정보가 기록된 정보저장매체, 그 재생 장치 및 재생 방법{Information storage medium containing control information for controlling buffering status of markup document, reproducing method and apparatus therefor}

**【도면의 간단한 설명】**

도 1은 AV 데이터가 기록된 인터랙티브 DVD의 개략도,

도 2는 도 1의 인터랙티브 DVD를 재생하는 과정에서 발생될 수 있는 끊김현상을 설명하기 위한 참고도,

도 3은 프리로드 또는 삭제를 수행하는 재생 장치의 블록도,

도 4는 프리로드 또는 삭제를 지원하는 DVD(300)의 디렉토리 구조를 도시한 참고도,

도 5는 프리로드 또는 삭제를 지원하는 DVD(300)의 볼륨 공간(Volume Space)의 개략도,

도 6은 프리로드 또는 삭제가 수행되는 과정을 설명하기 위한 플로우챠트,

도 7은 도 6의 프리로드 정보 해석단계(602단계)의 일 구현예,

도 8은 도 6의 프리로드 대상 파일의 프리로드 실행단계(603단계)의 일 구현예,

도 9a는 도 6의 프리로드 대상 파일의 프리로드 실행단계(603단계)의 다른 구현예,

도 9b는 도 6의 프리로드 대상 파일의 프리로드 실행단계(603단계)의 또 다른 구현예,

도 10은 메모리에 저장된 프리로드 대상 파일 중 적어도 하나를 삭제하는 과정을 설명하기 위한 플로우챠트,

도 11은 도 10의 삭제 실행단계(1002단계)의 일 구현예,

도 12는 도 1과 동일한 순서로 AV 데이터 및 HTML 문서가 기록되어 있을 때 본 발명에 따른 프리로드 수행에 의한 효과를 설명하기 위한 참고도,

도 13 및 14는 본 발명의 바람직한 실시예에 따른 재생 장치의 블록도,

도 15는 본 발명에 따라 버퍼 매니저가 ENAV 버퍼의 마크업 문서의 버퍼링 상태를 매니징하는 과정을 보여주는 개념도,

도 16은 본 발명의 바람직한 실시예에 따라 컨텐트 디코더와 버퍼 매니저에 의해 실행되는 버퍼 상태 제어 방법을 설명하는 플로우챠트이다.

도 17은 본 발명의 바람직한 실시예에 따라 AV 데이터와 마크업 문서가 기록된 디스크의 개략도,

도 18은 도 17과 같이 AV 데이터와 마크업 문서가 기록된 디스크의 디렉토리 구조도,

도 19는 도 17에 따른 디스크의 블롭 구조와 파일 구조를 보여주는 구조도,

도 20은 본 발명에 따라 도 17의 디스크에 저장된 마크업 문서 데이터와 AV 데이터가 재생되는 과정을 보여준다.

### 【발명의 상세한 설명】

#### 【발명의 목적】

#### 【발명이 속하는 기술분야 및 그 분야의 종래기술】

<21> 본 발명은 마크업 문서의 버퍼 상태를 제어하기 위한 정보가 기록된 정보저장매체, 그 재생 장치 및 재생 방법에 관한 것이다.

<22> 컨텐트와 함께 마크업 문서가 기록된 DVD(이하 "인터렉티브 DVD"라 함)가 판매되고 있다. 인터렉티브 DVD에 기록된 컨텐트는 두 가지 방법으로 재생될 수 있다. 하나는 일반 DVD와 동

일하게 디스플레이되는 비디오 모드이고, 다른 하나는 마크업 문서에 의해 정의되는 표시창을 통해 디스플레이되는 인터랙티브 모드이다. 사용자에 의해 인터랙티브 모드가 선택되면 PC에 탑재된 웹브라우저는 인터랙티브 DVD에 기록되어 있는 마크업 문서를 디스플레이한다. 마크업 문서에 의해 정의되는 표시창에는 사용자가 선택한 컨텐트가 표시된다.

<23> 예를 들어, 컨텐트가 AV 데이터인 영화일 경우 마크업 문서의 표시창에는 영화가 상영되고 나머지 부분에는 대본, 줄거리, 출연배우의 사진, 등 다양한 부가정보가 표시될 수 있다. 부가정보는 이미지파일 또는 텍스트파일을 포함한다.

<24> 도 1은 AV 데이터가 기록된 인터랙티브 DVD의 개략도이다.

<25> 도 1을 참조하면, 인터랙티브 DVD의 트랙에는 AV 데이터가 MPEG 비트스트림으로 기록되어 있고 복수개의 마크업 문서가 기록되어 있다. 여기서, 마크업 문서는 마크업 문서에 삽입되는 다양한 이미지 파일, 그래픽 파일 등을 망라한 마크업 리소스(resource)를 의미한다.

<26> 도 2는 도 1의 인터랙티브 DVD를 재생하는 과정에서 발생될 수 있는 끊김현상을 설명하기 위한 참고도이다.

<27> 도 2를 참조하면, AV 데이터가 버퍼링되는 버퍼 메모리의 상태(occupancy)와 웹 리소스가 캐싱되는 캐시 메모리의 상태가 표시되어 있다. 도 1과 도 2를 참조하여 AV 데이터가 메모리에 로드되어 디스플레이되는 과정을 살펴보면, 픽업장치는 STARTUP.HTM을 탐색하고 독출하여 캐시 메모리에 로드한다. 로드된 STARTUP.HTM은 활성화된다. 이와 동시에 사용자의 선택에 따른 ① AV 데이터가 버퍼 메모리로 로드된 다음 디스플레이되기 시작한다. 이어서 ② AV 데이터가 로드되어 디스플레이된다. ② AV 데이터의 버퍼링이 완료되면 픽업장치는 ③ AV 데이터가 기록된 위치로 점프하여 버퍼링을 시작한다. 이 때, 사용자가 ④ A.HTM를 요구하면 픽업

장치는 ③ AV 데이터에 대한 버퍼링을 중단하고 ④ A.HTM을 탐색하여 캐시 메모리로 로드한다. 그 동안에도 ③ AV 데이터는 계속 디스플레이되기 때문에 버퍼 메모리에 로드되어 디스플레이 가능한 데이터는 급격히 줄어들게 된다. ④ A.HTM이 활성화되고 ③ AV 데이터의 버퍼링이 완료된 다음 이어서 ⑤ AV 데이터를 버퍼링한다. ⑤ AV 데이터의 버퍼링이 완료되면 픽업장치는 ⑥ AV 데이터가 기록된 위치로 점프한다. 이와 같은 경우, 버퍼링되어 있던 데이터가 모두 소진되는 현상이 발생할 수 있다. 즉, 종래 인터랙티브-DVD에 있어서, DVD-Video의 영상과 마크업 문서가 서로 동기되어 보여져야 할 경우(예: 특정 배우가 등장하면 그 배우에 대한 약력이 표시되는 경우) 픽업장치는 AV 데이터의 버퍼링을 중단하고 대응 마크업 문서를 탐색하여 캐싱해야 하기 때문에 영상이 갑자기 끊어지는 현상이 발생될 수 있다.

<28> 이에, 본 출원인은 2002년 9월 19일자로 한국특허출원 제02-57393호 "프리로드 정보가 기록된 정보저장매체, 그 재생 장치 및 재생 방법"을 출원한 바 있다. 여기에는 마크업 문서를 사용하여 소정 컨텐트를 인터랙티브 모드로 재생함에 있어서 컨텐트의 끊김현상이 발생되지 않도록 해주는 <프리로드>가 개시되어 있다. 다만, <프리로드>에 의하면, 프리로드 대상 파일 전체를 프리로드할 경우에는 문제가 없으나, 전체 프리로드 대상 파일 중, 적어도 일부를 제대로 읽어 들이지 못할 경우 모두 프리젠테이션할 수 없는 상황이 발생할 수 있다.

#### 【발명이 이루고자 하는 기술적 과제】

<29> 따라서, 본 발명의 목적은 프리로드를 수행함에 있어 프리로드 대상 파일 중 적어도 일부를 제대로 프리로드하지 못할 경우라하더라도 적절히 프리젠테이션이 가능하도록 마크업 문서의 버퍼링 상태를 제어할 수 있는 제어 정보가 기록된 정보저장매체, 그 재생 장치 및 재생 방법을 제공함에 있다.

### 【발명의 구성 및 작용】

<30> 상기 목적은 본 발명에 따라, AV 데이터를 마크업 문서를 사용하여 인터랙티브 모드로 재생하는 재생 장치에 있어서, 상기 마크업 문서를 버퍼링하는 버퍼; 및 상기 버퍼를 관리하여 상기 마크업 문서를 프리로드하는 버퍼 매니저를 포함하고, 상기 버퍼 매니저는 report 신호에 응답하여 상기 버퍼의 버퍼링 상태 정보를 출력하는 것을 특징으로 하는 재생 장치에 의해서 달성된다.

<31> 상기 재생 장치는 상기 마크업 문서를 해석하고 디코딩하며 상기 버퍼 매니저로 상기 report 신호를 발생시키는 컨텐트 디코더를 더 포함하고, 상기 버퍼 매니저는 상기 버퍼링 상태 정보를 상기 컨텐트 디코더로 알려주는 것이 바람직하다.

<32> 상기 컨텐트 디코더는 API를 이용하여 상기 report 신호를 발생시키거나, 상기 마크업 문서의 파일 경로 및 상기 마크업 문서의 속성 중 적어도 하나를 매개변수로 갖는 API를 이용하여 상기 report 신호를 발생시키거나, [obj].isCached(URL, resType) API를 이용하여 상기 report 신호를 발생시킴이 바람직하다(여기서, URL은 상기 마크업 문서의 파일 경로를 나타내는 매개변수이고, resType은 상기 마크업 문서의 속성을 나타내는 매개변수이다).

<33> 상기 컨텐트 디코더는 API를 통해 상기 버퍼 매니저가 대응하는 마크업 문서의 프리로드에 성공하였는지, 실패하였는지 또는 읽는 중인지를 통지받는 것이 바람직하다.

<34> 상기 API는 상기 버퍼 매니저가 대응하는 마크업 문서의 읽기에 성공하면 0 을 반환하고, 실패하면 1을 반환하며, 읽는 중이면 2를 반환하는 것이 바람직하다.

<35> 상기 버퍼 매니저는 상기 버퍼링 상태 정보를 API를 통해 상기 컨텐트 디코더로 알려주는 것이 바람직하다.

<36> 한편, 본 발명의 다른 분야에 따르면, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법에 있어서, (a) 상기 마크업 문서를 프리로드하기 위해 버퍼링하는 단계; 및 (b) report 신호에 응답하여 상기 마크업 문서의 버퍼링 상태 정보를 출력하는 단계를 포함하는 것을 특징으로 하는 재생 방법에 의해서도 달성된다.

<37> 상기 (b)단계는 (b1) 상기 마크업 문서의 버퍼링을 관리하는 버퍼 매니저로 상기 마크업 문서를 해석하고 디코딩하는 컨텐트 디코더가 상기 report 신호를 발생시키는 단계; 및 (b2) 상기 버퍼 매니저가 상기 버퍼링 상태 정보를 상기 컨텐트 디코더로 알려주는 단계를 포함하는 것이 바람직하다.

<38> 상기 (b1)단계는 상기 컨텐트 디코더가 API를 이용하여 상기 report 신호를 발생시키는 단계이거나, 상기 컨텐트 디코더가 상기 마크업 문서의 파일 경로 및 상기 마크업 문서의 속성 중 적어도 하나를 매개변수로 갖는 API를 이용하여 상기 report 신호를 발생시키는 단계이거나, 상기 컨텐트 디코더가 [obj].isCached(URL,resType) API를 이용하여 상기 report 신호를 발생시키는 단계임이 바람직하다(여기서, URL은 상기 마크업 문서의 파일 경로를 나타내는 매개 변수이고, resType은 상기 마크업 문서의 속성을 나타내는 매개변수이다).

<39> 상기 (b2)단계는 상기 컨텐트 디코더가 API를 통해 상기 버퍼 매니저로부터 대응하는 마크업 문서의 읽기에 성공하였는지, 실패하였는지 또는 읽는 중인지를 상기 버퍼링 상태 정보로서 통지받는 단계이거나, 상기 API는 상기 버퍼 매니저가 대응하는 마크업 문서의 읽기에 성공하면 상기 버퍼링 상태 정보로서, 0을 반환하고, 실패하면 1을 반환하며, 읽는 중이면 2를 반환하는 단계를 포함하는 것이 바람직하다.

<40> 한편, 본 발명의 다른 분야에 따르면, AV 데이터; 상기 AV 데이터를 인터랙티브 모드로 재생하기 위한 복수개의 마크업 문서; 및 상기 AV 데이터를 인터랙티브 모드로 재생하기 위해

필요한 마크업 문서를 프리로드하기 위한 버퍼링 상태 정보를 알아보기 위한 제어 정보를 포함하는 것을 특징으로 하는 정보저장매체에 의해서도 달성된다.

<41> 상기 제어 정보는 소정 마크업 문서에 기록되어 있거나, 상기 버퍼링 상태 정보를 알아보기 위한 report 신호를 발생시키는 API를 사용하여 기록되거나, 상기 버퍼링 상태 정보를 알아보기 위한 report 신호를 발생시키는 [obj].isCached(URL, resType) API를 사용하여 기록됨이 바람직하다(여기서, URL은 상기 마크업 문서의 파일 경로를 나타내는 매개변수이고, resType은 상기 마크업 문서의 속성을 나타내는 매개변수이다).

<42> 또한, 본 발명의 다른 분야에 따르면, 상기 목적은 AV 데이터를 마크업 문서를 사용하여 인터랙티브 모드로 재생하는 재생 장치에 있어서, 상기 마크업 문서를 버퍼링하는 버퍼; 상기 버퍼를 관리하여 상기 마크업 문서를 상기 버퍼에 프리로드하는 버퍼 매니저; 및 상기 버퍼에 버퍼링되어 있는 마크업 문서를 가져와서 해석하고 디코딩하는 컨텐트 디코더를 포함하고, 상기 버퍼 매니저는 상기 버퍼의 프리로드 완료 여부를 상기 컨텐트 디코더로 알려주는 것을 특징으로 하는 재생 장치에 의해서도 달성된다.

<43> 상기 버퍼 매니저는 상기 버퍼의 프리로드 완료 여부를 API를 사용하여 상기 컨텐트 디코더로 알려주거나, fetch 신호에 응답하여 대응하는 마크업 문서를 상기 버퍼에 프리로드하거나, 상기 컨텐트 디코더로부터의 fetch 신호에 응답하여 대응하는 마크업 문서를 상기 버퍼에 프리로드하거나, 상기 컨텐트 디코더로부터의 fetch 신호에 의한 프리로드의 완료 여부를 상기 컨텐트 디코더로 알려주는 것이 바람직하다.

<44> 상기 컨텐트 디코더는 API를 이용하여 상기 fetch 신호를 발생시키거나, API를 이용하여 상기 버퍼 매니저로 상기 버퍼의 프리로드 완료 여부를 확인하거나, 대응하는 마크업 문서의

프리로드가 완료되었는지 여부를 API를 사용하여 확인하거나, 상기 프리로드의 완료 여부를 [obj].allDone API를 사용하여 확인하는 것이 바람직하다.

<45> 상기 [obj].allDone API는 현재 성공 완료이면 성공을 반환하고, 그렇지 않으면 실패를 반환하는 것이 더욱 바람직하다.

<46> 한편, 본 발명의 다른 분야에 따르면, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법에 있어서, (a) 상기 마크업 문서를 프리로드하기 위해 버퍼링하는 단계; 및 (b) 상기 마크업 문서의 버퍼링이 완료되었는지 여부에 대한 정보를 출력하는 단계를 포함하는 것을 특징으로 하는 재생 방법에 의해서도 달성된다.

<47> 상기 (a)단계는 프리로드를 지시하는 fetch 신호에 응답하여 버퍼링하는 단계를 포함하는 것이 바람직하다.

<48> 상기 (b)단계는 상기 마크업 문서의 버퍼링이 완료되었는지 여부를 확인하는 API에 응답하여 수행되거나, 상기 마크업 문서의 버퍼링이 완료되었는지 여부를 확인하는 [obj].allDone API에 응답하여 수행됨이 바람직하다.

<49> 한편, 본 발명의 다른 분야에 따르면, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법에 있어서, (a) fetch 신호를 사용하여 상기 마크업 문서의 프리로드를 지시하는 단계; 및 (b) 상기 fetch 신호에 의한 프리로드 지시의 성공 여부를 응답받는 단계를 포함하는 것을 특징으로 하는 재생 방법에 의해서도 달성된다.

<50> 상기 (a)단계는 컨텐트 디코더가 API를 이용하여 상기 fetch 신호를 상기 마크업 문서의 프리로드를 수행하는 버퍼 매니저로 발생시키는 단계임이 바람직하다.

<51> 상기 (b)단계는 상기 마크업 문서의 프리로드를 수행하는 버퍼 매니저가 상기 프리로드 지시의 성공 여부를 API를 이용하여 상기 컨텐트 디코더로 알려주는 단계임이 바람직하다.

<52> 또한, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법에 있어서, (a) 컨텐트 디코더가 API를 이용하여 현재 프리로드를 완료하였는지 여부를 버퍼 매니저로 확인하는 단계; 및 (b) 상기 버퍼 매니저는 상기 API를 이용하여 현재 프리로드 작업을 성공 완료하였으면 상기 컨텐트 디코더로 성공을 반환하고, 그렇지 않으면 상기 컨텐트 디코더로 실패를 반환하는 단계를 포함하는 것을 특징으로 하는 재생 방법에 의해서도 달성된다.

<53> 상기 API는 [obj].allDone API임이 바람직하다.

<54> 한편, 본 발명의 다른 분야에 따르면, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법을 수행하기 위한, 컴퓨터에 의해 실행가능한 프로그램 코드가 기록된 정보저장매체에 있어서, 상기 마크업 문서를 프리로드하기 위한 버퍼링을 수행하기 위한 제1 프로그램 코드; 및 상기 마크업 문서의 버퍼링이 완료되었는지 여부에 대한 정보를 출력하기 위한 제2 프로그램 코드를 포함하는 것을 특징으로 하는 정보저장매체에 의해서도 달성된다.

<55> 상기 제1 프로그램 코드는 프리로드를 지시하는 fetch 신호에 응답하여 실행됨이 바람직하다.

<56> 상기 제2 프로그램 코드는 상기 마크업 문서의 버퍼링이 완료되었는지 여부를 확인하는 API에 응답하여 실행되거나, 상기 마크업 문서의 버퍼링이 완료되었는지 여부를 확인하는 [obj].allDone API에 응답하여 실행됨이 바람직하다.

<57> 또한, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법을 수행하기 위한, 컴퓨터에 의해 실행가능한 프로그램 코드가 기록된 정보저장매체에 있어서, fetch 신호를 사용하여 상기 마크업 문서의 프리로드를 지시하는 제1 프로그램 코드; 상기 fetch 신호에 의한 프리로드 지시의 성공 여부를 응답받는 제2 프로그램 코드를 포함하는 것을 특징으로 하는 정보저장매체에 의해서도 달성된다.

<58> 상기 제1 프로그램 코드는 컨텐트 디코더가 API를 이용하여 상기 fetch 신호를 상기 마크업 문서의 프리로드를 수행하는 버퍼 매니저로 발생시키기 위한 것임이 바람직하다.

<59> 상기 제2 프로그램 코드는 상기 마크업 문서의 프리로드를 수행하는 버퍼 매니저가 상기 프리로드 지시의 성공 여부를 API를 이용하여 상기 컨텐트 디코더로 알려주기 위한 것임이 바람직하다.

<60> 또한, 상기 목적은 마크업 문서를 프리로드하여 AV 데이터를 인터랙티브 모드로 재생하는 방법을 수행하기 위한, 컴퓨터에 의해 실행가능한 프로그램 코드가 기록된 정보저장매체에 있어서, 컨텐트 디코더가 API를 이용하여 현재 프리로드를 완료하였는지 여부를 버퍼 매니저로 확인하기 위한 제1 프로그램 코드; 및 상기 버퍼 매니저는 상기 API를 이용하여 현재 프리로드 작업을 성공 완료하였으면 상기 컨텐트 디코더로 성공을 반환하고, 그렇지 않으면 상기 컨텐트 디코더로 실패를 반환하도록 하는 제2 프로그램 코드를 포함하는 것을 특징으로 하는 정보저장매체에 의해서도 달성된다.

<61> 상기 API는 [obj].allDone API임이 바람직하다.

<62> 이하 첨부된 도면을 참조하여 본 발명의 바람직한 실시예를 상세히 설명한다.

<63> 이해를 돋기 위해, 먼저 <프리로드> 및 <삭제>를 수행하는 재생 장치 및 그 방법에 대해 개략적으로 설명한다.

<64> 도 3은 프리로드 또는 삭제를 수행하는 재생 장치의 블록도이다.

<65> 도 3을 참조하면, 재생 장치는 DVD(300)에 기록된 AV 데이터를 디코딩하여 AV 데이터 스트림으로 재생한 다음 마크업 문서에 의해 정의되는 표시창을 통해 디스플레이할 수 있는 인터렉티브 모드를 지원하는 장치로서, 독출부(1), 제1 메모리(2), 제2 메모리(3), AV 디코더(4), 및 프리젠테이션 엔진(5)을 포함한다. 인터렉티브 모드에서 디스플레이되는 화면은 인터렉티브 화면으로, 마크업 화면에 AV 화면이 매립된 화면을 가리킨다. 마크업 화면은 마크업 문서가 디스플레이된 화면이고 AV 화면은 AV 데이터가 재생되어 얻어지는 화면을 말한다.

<66> 프리젠테이션 엔진(5)은 후술하는 바와 같이 링크 태그, 자바스크립트(JavaScript) API(Application Program Interface) 또는 자바 애플릿(Java applet) API로 구현된 프리로드 정보 또는 자바스크립트 API 또는 자바 애플릿 API로 구현된 삭제 정보가 해석되어 실행될 수 있도록 링크 태그의 확장, 자바스크립트, 또는 자바 애플릿을 지원한다.

<67> 독출부(1)는 DVD(300)로부터 마크업 문서 또는 AV 데이터를 독출한다. 제1 메모리(2)는 버퍼 메모리로서 독출부(1)에 의해 독출된 AV 데이터를 버퍼링한다. 제2 메모리(3)는 캐시 메모리로서 수신된 프리로드 파일을 캐싱한다. AV 디코더(4)는 제1 메모리(2)에 저장된 AV 데이터를 디코딩하여 AV 데이터 스트림을 출력한다. 프리젠테이션 엔진(5)은 독출된 마크업 문서에 포함된 프리로드 정보를 해석하고, 해석된 프리로드 정보에 따라 제2 메모리(3)에 프리로드 되도록 독출부(1) 또는 인터넷 서버(도시되지 않음)로 프리로드 대상 파일을 요청한다. 또한, 프리로드 대상 파일이 AV 데이터와 동기되어 디스플레이되어야 할 경우 그 프리로드 대상 파일을 제2 메모리(3)로부터 독출하여 AV 디코더(4)로부터 출력된 AV 데이터 스트림과 함께 디스플

레이되도록 한다. 또한, 삭제 정보를 해석하여 삭제 대상 파일을 제2 메모리(3)로부터 삭제한다.

<68> 본 실시예에 따른 DVD(300)에는 오디오 데이터 또는 비디오 데이터를 포함하는 AV 데이터가 기록되어 있는 한편, 프리로드 정보 및/또는 삭제 정보가 포함된 마크업 문서가 기록되어 있다. 나아가, 프리로드 리스트 파일 및/또는 삭제 리스트 파일이 더 기록될 수 있다.

<69> 프리로드 리스트 파일에는 프리로드되어야 할 파일의 명칭이 나열된 프리로드 리스트 및 각 프리로드 대상 파일을 저장하기 위해 필요한 메모리의 크기정보가 기록되어 있다. 프리로드 대상 파일은 대응 AV 데이터가 디스플레이될 경우 동기되어 재생되어야 할 마크업 문서로서 본 실시예에 따라 DVD(300)에 기록되어 있다. 다만, 프리로드 대상 파일은 인터넷을 통해 접속 가능한 인터넷 서버에 저장되어 있을 수 있다.

<70> 「프리로드 정보」는 프리로드 대상 파일을 독출하여 캐시 메모리에 저장할 것을 명령하는 정보이다. 프리로드 정보는 일 예로, 프리로드 리스트 파일의 경로 및/또는 속성이 삽입되어 있는 링크 태그로 구현될 수 있다. 이때 링크 태그는 헤드 태그 내에 삽입된다. 다른 예로, 프리로드 리스트 파일의 경로 및/또는 속성을 매개변수로 가지며 프리로드 리스트 파일을 호출하는 자바스크립트 API 또는 자바 애플릿 API로 구현된다. 또 다른 예로, 직접 프리로드 대상 파일의 경로 및/속성을 매개변수로 가지며 프리로드 대상 파일을 호출하는 자바스크립트 API 또는 자바 애플릿 API로 구현된다. 이 경우 프리로드 리스트 파일은 존재하지 않는다.

<71> 삭제 리스트 파일에는 삭제되어야 할 파일의 위치정보(파일명 및 경로)가 나열된 삭제 리스트가 기록되어 있다. 「삭제 정보」는 삭제 대상 파일을 제2 메모리(3)로부터 삭제할 것을 명령하는 정보이다. 삭제 정보는 일 예로 삭제 리스트 파일의 위치정보를 매개변수로 가지며 삭제 리스트 파일에 기록된 삭제 대상 파일을 삭제하는 자바스크립트 API 또는 자바 애플릿

API로 구현된다. 다른 예로, 직접 삭제 대상 파일의 경로 및/또는 속성을 매개변수로 가지며 삭제 대상 파일을 삭제하는 자바스크립트 API 또는 자바 애플릿 API로 구현된다. 이 경우 삭제 리스트 파일은 존재하지 않는다.

<72>        도 4는 프리로드 또는 삭제를 지원하는 DVD(300)의 디렉토리 구조를 도시한 참고도이다.

<73>        도 4를 참조하면, 루트 디렉토리에는 AV 데이터가 포함된 DVD 비디오 디렉토리 VIDEO\_TS 와 마크업 문서 등 인터랙티브 기능을 지원하기 위한 데이터가 기록된 DVD 인터랙티브 디렉토리 DVD\_ENAV가 마련되어 있다.

<74>        VIDEO\_TS에는 포함된 비디오 타이틀 전체에 대한 헤더 정보가 기록된 VIDEO\_TS.IFO 및 첫 번째 비디오 타이틀에 대한 네비게이션 정보가 기록된 VTS\_01\_0.IFO가 기록되어 있고, 이어서 비디오 타이틀을 구성하는 AV 데이터인 VTS\_01\_0.VOB, VTS\_01\_1.VOB, …이 기록되어 있다. 보다 상세한 구성은 DVD-Video 표준인 「DVD-Video for Read Only Memory Disc 1.0」에 개시되어 있다.

<75>        DVD\_ENAV에는 인터랙티브 정보 전체에 대한 네비게이션 정보가 기록된 DVD\_ENAV.IFO 가 기록되어 있고, 시작 문서로 지정된 STARTUP.HTM이 기록되어 있으며, 본 실시예에 따른 프리로드 리스트 파일인 STARTUP.PLD가 마련되어 있다. 또한, 프리로드 대상 파일인 A.HTM, 또는 A.HTM에 삽입되어 표시되기 위한 그래픽 파일 A.PNG가 기록되어 있다. 기타 프리로드 대상 파일 및 이에 삽입되어 표시되기 위한 다양한 형식의 파일이 기록될 수 있다.

<76>        도 5는 프리로드 또는 삭제를 지원하는 DVD(300)의 볼륨 공간(Volume Space)의 개략도이다.

<77> 도 5를 참조하면, DVD(300)의 Volume Space에는 Volume과 파일에 대한 제어 정보가 기록된 제어정보 영역, 대응 비디오 타이틀 데이터가 기록된 DVD-Video 데이터 영역, 및 인터랙티브 모드로 재생될 수 있도록 하는 DVD-Interactive Data 영역을 포함한다.

<78> DVD-Video 데이터 영역에는 도 4의 DVD 비디오 디렉토리 DVD\_TS에 저장된 파일, 즉 VIDEO\_TS.IFO, VTS\_01\_0.IFO, VTS\_01\_0.VOB, VTS\_01\_1.VOB, ··· 등이 기록되어 있다. DVD-Interactive 데이터 영역에는 도 4의 DVD 인터랙티브 디렉토리 DVD\_ENAV에 저장된 파일, 즉 STARTUP.HTM, STARTUP.PLD, A.HTM 및 A.PNG이 기록되어 있다.

<79> 상기와 같은 구성을 기초로 프리로드 또는 삭제가 수행되는 과정을 설명하면 다음과 같다.

<80> 도 6은 프리로드 또는 삭제가 수행되는 과정을 설명하기 위한 플로우챠트이다.

<81> 도 6을 참조하면, 인터랙티브 모드가 선택되면 독출부(1)는 DVD(300)에 기록되어 있는 마크업 문서로서 본 실시예에 따른 HTML 문서를 독출하고(601단계), 프리젠테이션 엔진(5)은 HTML 문서에 포함되어 있는 프리로드 정보를 해석하여 독출부(1) 또는 인터넷 서버로 프리로드를 요청한다(602단계). 이에, 프리로드 대상 파일이 캐시 메모리인 제2 메모리(3)에 저장된다(603단계).

<82> 한편, 독출부(1)는 대응 AV 데이터를 DVD(300)로부터 독출하여 버퍼 메모리인 제1 메모리(2)에 저장한다(604단계). AV 디코더(4)는 제1 메모리(1)에 저장된 AV 데이터를 디코딩하는 한편(605단계), 프리젠테이션 엔진(5)은 제2 메모리(3)로부터 프리로드 대상 파일을 독출하고 AV 디코더(4)에 의해 디코딩된 AV 데이터 스트림이 HTML 문서에 의해 정의된 표시창에 디스플레이되도록 한다(606단계).

<83> 도 7은 도 6의 프리로드 정보 해석단계(602단계)의 일 구현예이다.

<84> 도 7을 참조하면, 프리젠테이션 엔진(5)은 HTML 문서에 기록된 프리로드 리스트 파일의 경로를 인식하고(701단계), 인식된 경로로부터 프리로드 리스트 파일을 독출한다(702단계). 다음으로, 프리로드 리스트 파일에 기록된 프리로드 대상 파일을 인식한다(703단계). 여기서, 프리로드 대상 파일을 인식한다는 것은 프리로드 대상 파일의 경로, 나아가 속성을 파악하는 것을 의미한다.

<85> 도 8은 도 6의 프리로드 대상 파일의 프리로드 실행단계(603단계)의 일 구현예이다. 도 8을 참조하면, 프리젠테이션 엔진(5)은 HTML 문서의 헤드 태그 내에 있는 링크 태그 내에 기록된 프리로드 리스트 파일의 경로를 인식하여 프리로드 리스트 파일을 호출한다(801단계). 다음으로, 프리젠테이션 엔진(5)은 프리로드 대상 파일의 경로 및 속성을 변수로 갖는 프리로드 태그를 포함하는 프리로드 리스트 파일을 해석하여 프리로드를 실행한다(802단계).

<86> 도 9a는 도 6의 프리로드 대상 파일의 프리로드 실행단계(603단계)의 다른 구현예이다. 도 9a를 참조하면, 프리젠테이션 엔진(5)은 바디 태그 내에 삽입되어, 프리로드 리스트 파일의 경로를 매개변수로 갖는 API를 호출하여 프리로드 리스트 파일을 독출한다(901a단계). 다음으로, 프리젠테이션 엔진(5)은 프리로드 대상 파일의 경로 및 속성을 변수로 갖는 프리로드 태그를 포함하는 프리로드 리스트 파일을 해석하여 프리로드를 실행한다(901b단계).

<87> 도 9b는 도 6의 프리로드 대상 파일의 프리로드 실행단계(603단계)의 또 다른 구현예이다. 도 9b를 참조하면, 프리젠테이션 엔진(5)은 바디 태그 내에 삽입되어, 직접 프리로드 대상 파일의 경로 및 속성을 매개변수로 갖는 API를 호출하여 프리로드 대상 파일을 메모리에 저장한다(901b단계). 이때, 프리로드 대상 파일의 속성을 파악할 수 있으므로 프리젠테이션 엔진(5)은 속성에 따라 프리로드 대상 파일을 처리한 다음 메모리에 저장할 수 있다.

<88> 도 10은 메모리에 저장된 프리로드 대상 파일 중 적어도 하나를 삭제하는 과정을 설명하기 위한 플로우챠트이다.

<89> 도 10를 참조하면, 프리젠테이션 엔진(5)은 HTML 문서에 포함된 삭제정보를 해석하고 (1001단계), 삭제 리스트 파일로부터 인식된 삭제 대상 파일을 캐시 메모리인 제2 메모리(3)로부터 삭제한다(1002단계). 다만, 후술하는 소스코드에서 확인되듯이 본 실시예에서의 프리로드 리스트 파일과 삭제 리스트 파일은 동일한 하나의 파일, 즉 STARTUP.PLD로 구현하였다. 물론 프리로드 대상 파일의 리스트와 삭제 대상 파일의 리스트가 각각 분리되어 기록된 두 개의 파일로도 구현할 수 있다.

<90> 도 11은 도 10의 삭제 실행단계(1002단계)의 일 구현예이다.

<91> 도 11을 참조하면, 삭제 리스트 파일의 경로를 매개변수로 가지고 삭제 리스트 파일에 기록된 삭제 대상 파일을 캐시 메모리인 제2 메모리(3)로부터 삭제하는 API에 의해 실행되도록 한다(1101단계). 여기서, 「삭제」의 의미는 물리적으로 해당 데이터를 지우는 가비지 콜렉션뿐 아니라, 물리적으로 데이터는 그대로 둔 상태에서 플래그 등을 사용하여 해당 데이터가 삭제 가능함을 알리거나 여기에 다른 데이터를 기록할 수 있음을 알리는 것을 포함한다.

<92> 도 12는 도 1과 동일한 순서로 AV 데이터 및 HTML 문서가 기록되어 있을 때 본 발명에 따른 프리로드 수행에 의한 효과를 설명하기 위한 참고도이다.

<93> 도 12에는 MPEG 코딩된 AV 데이터가 버퍼링되는 제1 메모리(2)의 상태(occupancy)와 웹 리소스가 캐싱되는 제2 메모리(3)의 상태가 표시되어 있다. 도 1과 도 12를 참조하여 AV 데이터가 로드되어 디스플레이되는 과정을 살펴보면, 독출부(1)는 STARTUP.HTM을 탐색하여 독출하고 프리젠테이션 엔진(5)은 STARTUP.HTM에 삽입되어 있는 프리로드 정보를 해석하여 ④ A.htm

의 프리로드가 수행되도록 한다. 이에 ④ A.HTM이 제2 메모리(3)에 프리로드된다. 한편, 로드된 STARTUP.HTM은 활성화된다. 이와 동시에 사용자의 선택에 따른 ① AV 데이터가 제1 메모리(2)로 로드된 다음 디스플레이되기 시작한다. 이어서 ② AV 데이터가 로드되어 디스플레이된다. ② AV 데이터의 버퍼링이 완료되면 독출부(1)는 ③ AV 데이터를 찾아 점프하여 버퍼링을 시작한다. 이 때, 사용자가 ④ A.HTM를 요구하면 프리젠테이션 엔진(5)은 제2 메모리(3)에 프리로드되어 있는 ④ A.HTM을 독출하여 디스플레이한다. 즉, 독출부(1)가 ③ AV 데이터에 대한 버퍼링을 중단하고 ④ A.HTM을 DVD(300)로부터 탐색하여 제2 메모리(3)로 로드하는 작업을 수행하지 않아도 된다. 따라서, 독출부(1)는 버퍼링을 중단없이 수행할 수 있다. 다만, 독출부(1)가 ⑤ AV 데이터의 버퍼링을 마무리하고 ⑥ AV 데이터로 점프하는 과정에서 제1 메모리(2)에 버퍼링되어 있던 데이터가 들어들 수 있으나 이미 버퍼링해놓은 데이터의 양이 충분하기 때문에 데이터 부족현상은 발생되지 않는다. 즉, 인터랙티브 모드를 지원하는 DVD에 있어서 DVD-Video의 영상과 HTML 문서가 서로 동기되어 보여져야 할 경우에도(예: 특정 배우가 등장하면 그 배우에 대한 약력이 표시되는 경우) 대응 HTML 문서가 이미 제2 메모리(3)에 프리로드되어 있기 때문에 독출부(1)는 AV 데이터의 버퍼링을 중단하고 대응 HTML 문서를 탐색하여 캐싱해야할 필요가 없다.

<94> <프리로드> 및 프리로드된 데이터의 <삭제>에 대한 보다 상세한 설명은 본 출원인이 2002년 9월 19일자로 출원한 한국출원 제02-57393호 "프리로드 정보가 기록된 정보저장매체, 그 재생 장치 및 재생방법"에 기재되어 있다.

<95> 이하에서는, 본 발명에 따라, 프리로드되는 대상 파일들 중 적어도 일부를 제대로 읽어 들이지 못한 경우라도 적절한 수준에서 프리젠테이션할 수 있도록 마크업 문서의 버퍼링 상태를 적절히 제어할 수 있는 재생 장치, 방법 및 그 정보저장매체를 설명한다.

<96>        도 13은 본 발명의 바람직한 실시예에 따른 재생 장치의 블록도이다.

<97>        도 13을 참조하면, 재생 장치는 도 3의 재생 장치와 마찬가지로 인터랙티브 모드를 지원하는 장치로서, <프리로드>를 수행하여, 특히 본 발명에 따라 마크업 문서의 버퍼 상태를 관리하기 위해, AV 버퍼(20), AV 재생 엔진(40), ENAV 버퍼(30) 및 ENAV 엔진(50)을 포함한다.

<98>        AV 버퍼(20)는 도 3의 제1 메모리(2)에 대응하는 것으로, 디스크(100) 또는 네트워크로부터 가져온 AV 데이터를 버퍼링한다. AV 재생 엔진(40)은 AV 버퍼(20)에 버퍼링되어 있는 AV 데이터를 디코딩하여 AV 스트림을 출력한다. ENAV 버퍼(30)는 도 3의 제2 메모리(3)에 대응하는 캐시 메모리로서, 디스크(100) 또는 네트워크로부터 가져온 마크업 문서를 버퍼링한다. ENAV 엔진(50)은 도 3의 프리젠테이션 엔진(5)에 해당하는 것으로, <프리로드>를 수행하고, ENAV 버퍼(30)에 버퍼링되고 있는 마크업 문서의 버퍼링 상태를 제어한다. 나아가, ENAV 버퍼(30)에 저장되어 있는 마크업 문서를 해석하거나 디코딩하여 AV 재생 엔진(40)으로부터 출력된 AV 스트림이 인터랙티브 모드로 재생되도록 한다.

<99>        도 14는 도 13의 ENAV 엔진(50)의 상세 블록도이다.

<100>        도 14를 참조하면, ENAV 엔진(50)은 그 내부에 ENAV 버퍼(30)를 제어하는 버퍼 매니저(51)와 마크업 문서들을 해석하는 컨텐트 디코더(52)를 가진다.

<101>        컨텐트 디코더(52)는 마크업 문서를 파싱하여 해석하는 해석 엔진 및 네트워크로부터 마크업 문서를 가져오는 브라우저를 구비한다. 마크업 문서란 HTML, CSS, JAVASCRIPT와 같은 마크업 텍스트 데이터와 이미지 데이터나 오디오 데이터, 자바 프로그램 등과 같이 마크업 문서가 참조하는 바이너리 데이터를 망라하는 마크업 리소스를 말한다. 마크업 문서는 ENAV 엔진(50) 내의 버퍼 매니저(51)에 의해 디스크(100) 또는 네트워크로부터 불러들여진다.

<102> 버퍼 매니저(51)는 마크업 문서를 <프리로드>하거나 <삭제>함에 있어 본 발명에 따라 ENAV 버퍼(30)에 의한 마크업 문서의 버퍼링 상태를 매니징한다. 보다 구체적으로, 버퍼 매니저(51)는 컨텐트 디코더(52)로부터 입력되는 5 종류의 신호에 따라 동작하게 된다.

<103> 도 15는 본 발명에 따라 버퍼 매니저(51)가 ENAV 버퍼(30)의 마크업 문서의 버퍼링 상태를 매니징하는 과정을 보여주는 개념도이다.

<104> 도 15를 참조하면, 컨텐트 디코더(52)로부터 버퍼 매니저(51)로 입력되는 신호는 1) fetch 신호, 2) retrieve 신호, 3) release 신호, 4) discard 신호, 5) report 신호가 있다.

<105> 1) fetch 신호는 사용할 마크업 문서를 ENAV 버퍼(30)에 프리로드해 놓으라는 신호이다. 만약 지정한 마크업 문서가 이미 ENAV 버퍼(30)에 존재하면 I/O 매니저를 이용하여 데이터를 디스크(100)나 네트워크로부터 읽어들이지 않는다. I/O 매니저는 디스크(100)로부터 데이터를 읽어들이는 리더(도시되지 않음) 또는 네트워크로부터 데이터를 가져오기 위한 네트워크 데이터 송수신부(도시되지 않음)를 가리킨다. 리더는 디스크(100)로부터 파일을 독출하고, 네트워크 데이터 송수신부는 HTTP 프로토콜을 사용하여 네트워크로부터 소정 데이터를 가져오거나 내보낸다.

<106> 도 15를 참조하면, I/O 매니저는 HTTP 요구인 경우 unblocked I/O를 하고 디스크 상의 파일 요구는 blocked I/O를 사용한다고 정하고 있다. 네트워크로부터 마크업 문서를 가져오는 경우 한 번에 복수개의 마크업 문서를 동시에 가져오는 unblocked 방식에 의한다. 디스크(100)로부터 마크업 문서를 읽어들일 경우 한 번에 복수개의 마크업 문서를 읽어들이게 되면 리더에 구비된 꽉업부(도시되지 않음)가 마크업 문서가 기록된 위치로 여러번 이동해야 하므로 읽기 속도의 저하가 발생하므로, 디스크(100)에 저장된 마크업 문서를 읽어들일 경우 일 마크

업 문서를 다 읽고난 이후에 다음 마크업 문서를 읽어들이는 순차적인 blocked I/O 방식을 택 한다.

<107> 2) retrieve 신호는 ENAV 버퍼(30)로부터 컨텐트 디코더(52)로 데이터를 가지고 올 것을 요청하는 신호이다. 따라서 데이터가 디스크로부터 읽어들여지는 중이거나 네트워크로부터 다운로드되고 있는 중이라면 컨텐트는 데이터를 모두 가지고 올 때까지 멈추어(blocking) 있다

<108> 3) release 신호는 retrieve 신호를 이용하여 가지고 온 데이터를 사용하지 않을 것이다. 따라서 어떤 마크업 문서가 5 회에 걸쳐서 retrieve 신호에 의해 참조되었다면 release 신호도 5 회에 걸쳐서 발생하게 된다. retrieve 신호가 발생될 때마다 카운터가 올라간다면 release 신호가 발생될 때마다 카운터가 내려간다. 마크업 문서에 대응하는 카운터가 0가 되면(retrieve된 마크업 문서들이 모두 release되면) 후술하는 바와 같이 discard 신호가 발생되었을 때 곧바로 ENAV 버퍼(30)에서 삭제할 수 있게 된다.

<109> 4) discard 신호는 ENAV 버퍼(30)로부터 마크업 문서를 더 이상 사용하지 않을 것이므로 삭제해도 됨을 의미한다. 따라서 이 신호에 따라 지정된 마크업 문서 데이터를 ENAV 버퍼(30)에서 버려지게(메모리 공간을 차지하지 않음) 된다. 다만, 일어나는 어플리케이션에 의해 discard 신호가 발생하였더라도 다른 어플리케이션에 의해 해당 마크업 문서에 대해 retrieve 신호가 발생한 다음 대응 release 신호가 발생하지 않았다면 release 신호가 발생하여야만 대응 마크업 문서를 ENAV 버퍼(30)에서 삭제한다.

<110> 5) report 신호는 fetch 신호에 따라 읽어들인 마크업 문서가 제대로 ENAV 버퍼(30)에 로드되었는지? 아니면 오류로 인해 적어도 일부의 마크업 문서를 읽어들이지 못했는지? 아니면 지금 마크업 문서를 읽어들이고 있는 중인지를 확인해보

기 위한 신호이다. 또한, 현재 프리로드 진행 중인 마크업 문서의 위치(파일 이름), 지금까지 프리로드된 양, 앞으로 프리로드해야 할 양, 현재 프리로드 작업에 할당된 전체 마크업 문서들의 프리로드 양, 앞으로 프리로드해야 할 남아 있는 마크업 파일들의 양을 확인해 보기 위한 신호이다.

<111> 위의 신호를 발생시키기 위해 마크업 문서에 기록되어 있는 스크립트의 API는 다음과 같다.

<112> 1. [obj].preload(URL, resType)

<113> 1) 내용:

<114> 소정 프리로드 대상 파일들을 ENAV 버퍼(30)로 미리 읽어올 것을 지시하는 API이다. 사용되는 매개변수는 프리로드 리스트 파일, 프리로드 대상 파일의 위치정보, 나아가 프리로드 대상 파일의 속성을 나타낼 수 있다. 이 API는 fetch 신호를 발생시킨다. 이 함수는 디스크 (disc://)에서 또는 네트워크(http://)에서 읽혀지는 모든 파일에 대해서 적용이 가능하다.

<115> 2) 매개 변수:

<116> URL = : 프리로드 리스트 파일의 경로 또는 프리로드 대상 파일의 경로

<117> resType = : 대상 파일의 속성

<118> 3) 반환값:

<119> 프리로드 지시에 성공하면 0을 반환하고 실패하면 -1를 반환한다.

<120> 4) 예:

<121> navigator.preload("disc://dvd\_enav/a.htm", "text/xml") // disc에 있는 "disc://dvd\_enav/a.htm"의 프리로드 대상 파일을 읽어와라. 그 파일은 텍스트 파일로서 xml파일이다.

<122> navigator.preload("disc://dvd\_enav/a.pld", "xml/preload") // 디스크에 있는 "disc://dvd\_enav/a.pld"의 프리로드 리스트 파일에 참조되어진 대상 파일을 읽어와라. 그 파일은 프리로드 파일로서 xml파일이다.

<123> 2. [obj].discard(URL, resType)

<124> 1) 내용

<125> 지시하는 삭제 대상 파일들을 ENAV 버퍼(30)에서 삭제하는 API이다. 매개변수는 삭제 리스트 파일, 삭제 대상 파일의 위치정보, 및 삭제 대상파일의 속성을 나타낸다. 이 API는 discard 신호를 발생 시킨다.

<126> 2) 매개 변수:

<127> URL = : 삭제 리스트 파일의 경로 또는 삭제 대상 파일의 경로

<128> resType = : 대상 파일의 속성

<129> 3) 반환값:

<130> 삭제에 성공하면 0를 반환하고 실패하면 -1를 반환한다.

<131> 4) 예:

<132> navigator.discard("disc://dvd\_enav/a.htm", "text/xml") // disc에 있는 "disc://dvd\_enav/a.htm"의 삭제 대상 파일이 ENAV 버퍼(30)에 있으면 이를 삭제한다. 그 파일은 text파일로서 xml파일이다.

<133> navigator.discard("disc://dvd\_enav/a.pld", "xml/preload") // disc에 있는 "disc://dvd\_enav/a.pld"의 리스트 파일이 참조하는 파일을 캐시 메모리에서 삭제한다. 그 파일은 리스트 파일로서 xml 파일이다.

<134> 3. [obj].isCached(URL, resType)

<135> 1) 내용:

<136> 지시하는 조사 대상 파일들을 ENAV 버퍼(30)에서 완전하게 읽혀졌는지 확인하는 API이다. 매개변수는 조사 리스트 파일, 조사 대상 파일의 위치정보, 및 대상 파일의 속성을 나타낸다. 이 API는 report 신호를 발생시킨다. 이 함수는 디스크(disc://)에서 또는 네트워크 (http://)에서 읽혀지는 모든 파일에 대해서 적용이 가능하다.

<137> 2) 매개 변수:

<138> URL = : 조사 리스트 파일의 경로 또는 조사 대상 파일의 경로

<139> resType = : 대상 파일의 속성

<140> 3) 반환값:

<141> 조사 대상 파일 리스트 내의 또는 조사 대상 파일의 읽기에 성공하면 0을 반환하고 읽는 데 실패하면 1을 반환하며 읽는 파일 중 읽기를 실패한 파일이 없고 적어도 한 파일이 읽는 중이면 2을 반환한다.

<142> 4) 예:

<143> navigator.isCached("disc://dvd\_enav/a.htm", "text/xml") // disc에 있는 "disc://dvd\_enav/a.htm"를 읽는 작업이 완료 되었는지 조사한다. 그 파일은 text파일로서 xml 파일이다.

<144> navigator.isCached ("disc://dvd\_enav/a.pld","xml/preload") // disc에 있는 "disc://dvd\_enav/a.pld"의 리스트 파일이 참조하는 대상 파일들의 읽기 작업이 완료되었는지 조사한다. 그 파일은 리스트 파일로서 xml 파일이다.

<145> 4. [obj].progressNameOfFile

<146> 1) 내용:  
현재 프리로드 진행 중인 대상 파일의 URI를 리턴하는 속성 값.

<148> 2) 반환값:  
파일 경로명 또는 URI

<150> 5. [obj].progressLengthOfFile

<151> 1) 내용:  
현재 프리로드 진행 중인 대상 파일에서 현재까지 프리로드를 받은 양.

<153> 2) 반환값:  
바이트 단위로 표현된 양

<155> 6. [obj].remainLengthOfFile

<156> 1) 내용:  
현재 프리로드 진행 중인 대상 파일에서 현재까지 프리로드된 것을 제외하고 앞으로 프리로드해야 할 남은 양.

<158> 2) 반환값:  
바이트 단위로 표현된 양

<160> 7. [obj].totalLoadingSize

<161> 1) 내용:

<162> 현재 프리로드 파일에서 지정한 프리로드 대상 파일 크기의 전체 합계.

<163> 2) 반환값:

<164> 바이트 단위로 표현된 용량

<165> 8. [obj].remainLoadingSize

<166> 1) 내용:

<167> 현재 프리로드 파일에서 지정한 프리로드 대상 파일의 현재까지 프리로드된 것을 제외하고 앞으로 프리로드해야 할 남은 양 전체의 합계.

<168> 2) 반환값:

<169> 바이트 단위로 표현된 양

<170> 9. [obj].allDone

<171> 1) 내용:

<172> 현재 재생 장치의 프리로드 작업 완료 여부.

<173> 2) 반환값:

<174> 성공 완료하면 True, 실패 완료 또는 진행 중인 경우 False

<175> 상기 retrieve 신호나 release 신호는 해당 마크업 문서를 사용할 때마다 발생한다. 예를 들어 컨텐트 디코더(52)가 "disc://dvd\_enav/a.png"의 이미지를 프리젠테이션하고자 하는 경우 를 해석한 다음 retrieve 신호를 발생하여 버퍼 매니저(51)로 하여금 이미지를 가져오도록 한 다음 이미지가 디스플레이 장치(도시되지 않음)의

화면에 프리젠테이션한다. 마찬가지로, 컨텐트 디코더(52)는 디스플레이 화면에서 이미지의 프리젠테이션을 종료한 다음 release 신호를 발생시킨다.

<176>      도 16은 본 발명의 바람직한 실시예에 따라 컨텐트 디코더(52)와 버퍼 매니저(51)에 의해 실행되는 버퍼링 상태 제어 방법을 설명하는 플로우챠트이다.

<177>      도 17은 본 발명의 바람직한 실시예에 따라 AV 데이터와 마크업 문서가 기록된 디스크의 개략도이며, 도 18은 도 17과 같이 AV 데이터와 마크업 문서가 기록된 디스크의 디렉토리 구조를 보여주고, 도 19는 도 17에 따른 디스크의 볼륨 구조와 파일 구조를 보여준다. 도 20은 도 17의 디스크에 저장된 마크업 문서 데이터와 AV 데이터가 재생되는 순서를 보여준다.

<178>      STARTUP.HTM에서 A.HTM에서 D.HTM까지 심리스 재생하기 위해서는 STARTUP.PLD는 다음과 같이 구현되는 것이 바람직하다.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE PRELOAD PUBLIC "-//DVD//DTD DVD Preload List 1.0//EN"
"http://www.dvdforum.org/enav/dvd-preload-list.dtd"-->
<filedef type="text/xml" src="disc://dvd_enav//a.htm" />
<filedef type="text/xml" src="disc://dvd_enav//a.pld" />
<filedef type="image/png" src="dvd://dvd_enav//a1.png" />
<filedef type="image/png" src="dvd://dvd_enav//a2.png" />
<filedef type="image/png" src="dvd://dvd_enav//a3.png" />
<filedef type="text/xml" src="disc://dvd_enav//b.htm" />
<filedef type="text/xml" src="disc://dvd_enav//b.pld" />
<filedef type="audio/au" src="dvd://dvd_enav//b1.au" />
<filedef type="image/png" src="dvd://dvd_enav//b2.png" />
<filedef type="image/png" src="dvd://dvd_enav//b3.jpg" />
<filedef type="text/xml" src="disc://dvd_enav//c.htm" />
<filedef type="text/xml" src="disc://dvd_enav//c.pld" />
<filedef type="image/png" src="dvd://dvd_enav//c1.png" />
<filedef type="image/png" src="dvd://dvd_enav//c2.png" />
<filedef type="image/png" src="dvd://dvd_enav//c3.png" />
<filedef type="text/xml" src="disc://dvd_enav//d.htm" />
<filedef type="text/xml" src="disc://dvd_enav//d.pld" />
<filedef type="image/png" src="dvd://dvd_enav//d1.png" />
<filedef type="image/png" src="dvd://dvd_enav//d2.png" />
```

<180> 이와 같은 STARTUP.PLD를 이용하여 STARTUP.HTM이 처음 디스플레이 화면에 나타나면서  
인터넷브 브리젠테이션이 시작된다. 다음은 도 15에 도시된 개념에 따라 동작하기 위해 구  
현된 STARTUP.HTM의 일 예이다.

1020030058891

출력 일자: 2003/10/20

<181>

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//DVD/DTD XHTML DVD-HTML1.0//EN"
"http://www.dvdforum.org/enav/dvdhtml-1-0.dtd">
<html>
<head>
<title>WAR II STARTUP PAGE</title>
<script language="ecmascript">
<![CDATA[
function onload_handler ()
{
navigator.preload("disc://dvd_enav/startup.pld","xml/preload");
idplayer.subscribeToEvent(10);
idplayer.setTrigger(1,"00:30:35:00",1);
idplayer.play();
docbody.addEventListener("dvdevent",idplayer_handler,true);
}
function idplayer_handler(e)
{
switch(e.parm1)
{
case 10: .// trigger event
  if (e.parm2 == 1) // begin to die
  {
    while (navigator.isCached("disc://dvd_enav/a.pld","xml/preload") == 2
|| navigator.isCached("disc://dvd_enav/b.pld","xml/preload") == 2
|| navigator.isCached("disc://dvd_enav/c.pld","xml/preload") == 2
|| navigator.isCached("disc://dvd_enav/d.pld","xml/preload") == 2); // during
//reading;
    if (navigator.isCached("disc://dvd_enav/a.pld","xml/preload" ) == 1) // failed
    {
      idplayer.stop();
      location.href = "disc://dvd_enav/discerr.htm";
    }
    // to read c.pld is OK.
    location.href = "disc://dvd_enav/a.htm"; // jump to c.htm
  }
  break;
}
]]>
</script>
</head>
<body id="docbody" onload="onload_handler ()">
<object style="position: absolute; left: 150px; top: 100px; width: 370px; height: 250px"
data="dvd:video_ts" id="idplayer"/>


```

<182> 한편, 다음과 같은 B.HTM은 A.HTM에 이어지는 화면을 구성한다. B.HTM이 재생되는 중에, A.HTM의 프리젠테이션에 필요한 마크업 문서들을 모아 놓은, 즉 B.PLD 내에서 프리로드 대상 파일로 언급된 마크업 문서들은 ENAV 버퍼(30)로부터 제거된다. B.HTM에는 A.HTM의 프리젠테이션에 필요한 마크업 문서들을 모아 놓은, 즉 A.PLD 내에서 프리로드 대상 파일로 언급된 마크업 문서들은 ENAV 버퍼(30)로부터 제거할 것을 명령하는 코드가 기록되어 있다.

```
<183> <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//DVD/DTD XHTML DVD-HTML1.0//EN"
"http://www.dvdforum.org/enav/dvdhtml-1-0.dtd">
<html>
<head>
<title>WAR II B.HTM PAGE</title>
<script language="ecmascript">
<![CDATA[
function onload_handler ()
{
navigator.discard ("disc://dvd_enav/a.pld","xml/preload"); // any longer to use A.HTM
idplayer.subscribeToEvent(10)
idplayer.setTrigger(1,"50:35:00",1);
docbody.addEventListener("dvdevent", idplayer_handler, true);
}
function idplayer_handler(e)
{
switch(e.parm1)
{
case 10: .// trigger event
  if (e.parm2 == 1) // begin combat
  {
    while (navigator.isCached("disc://dvd_enav/c.pld","xml/preload") == 2); // during
//reading;
    if (navigator.isCached("disc://dvd_enav/c.pld","xml/preload" ) == 1) // failed
    {
      idplayer.stop();
      location.href = "disc://dvd_enav/discerr.htm";
    }
    // to read a.pld is OK.
    location.href = "disc://dvd_enav/c.htm"; // jump to c.htm
  }
  break;
}
}
]]>
</script>
</head>
<body id="docbody" onload="onload_handler ()">
<object style="left: 110px; top: 80px; width: 500px; height: 200px" data="dvd:video_ts" id="idplayer"/>



</body>
```

<184> 한편, 전술한 인터랙티브 모드에서의 재생 방법은 컴퓨터 프로그램으로 작성 가능하다. 상기 프로그램을 구성하는 코드들 및 코드 세그먼트들은 당해 분야의 컴퓨터 프로그래머에 의하여 용이하게 추론될 수 있다. 또한, 상기 프로그램은 컴퓨터가 읽을 수 있는 정보저장매체 (computer readable medium)에 저장되고, 컴퓨터에 의하여 읽혀지고 실행됨으로써 상기 인터랙티브 모드에서 재생 방법을 구현한다. 상기 정보저장매체는 자기 기록매체, 광 기록매체, 및 캐리어 웨이브 매체를 포함한다.

#### 【발명의 효과】

<185> 전술한 바와 같이 본 발명에 따르면, 컨텐트의 제작시 프리로드된 상황을 알 수 있는 방법과 그 절차를 적용시킴으로써, 디스크의 물리적 파손이 발생하거나 연결 끊김 등 네트워크 오동작이 발생하여 프리로드 대상 파일의 전체가 프리로드되지 않고 적어도 일부만이 프리로드된 경우에도 프리로드된 파일들만으로의 프리젠테이션이 가능하게 함으로써 컨텐트를 재생함에 있어 발생할 수 있는 오류에 대한 재생 내구력을 높일 수 있다.

<186> 또한, 현재까지 얼마나 많은 양을 프리로드했는지, 앞으로 프리로드 과정이 완료되려면 얼마나 더 기다려야 하는지 스크립트 코드를 사용하여 계산할 수 있으므로, 컨텐트 제작자는 사용자에게 프리로드 진행 정보를 제공하여 사용자가 무작정 오랫동안 기다려야 하는 지루함을 없애줄 수 있다.

**【특허청구범위】****【청구항 1】**

AV 데이터를 마크업 문서를 사용하여 인터랙티브 모드로 재생하는 재생 장치에 있어서,

상기 마크업 문서를 버퍼링하는 버퍼;

상기 버퍼를 관리하여 상기 마크업 문서를 상기 버퍼에 프리로드하는 버퍼 매니저; 및

상기 버퍼에 버퍼링되어 있는 마크업 문서를 가져와서 해석하고 디코딩하는 컨텐트 디코더를 포함하고,

상기 버퍼 매니저는 상기 버퍼의 프리로드 완료 여부를 상기 컨텐트 디코더로 알려주는 것을 특징으로 하는 재생 장치.

**【청구항 2】**

제1항에 있어서,

상기 버퍼 매니저는 상기 버퍼의 프리로드 완료 여부를 API를 사용하여 상기 컨텐트 디코더로 알려주는 것을 특징으로 하는 재생 장치.

**【청구항 3】**

제1항에 있어서,

상기 버퍼 매니저는 fetch 신호에 응답하여 대응하는 마크업 문서를 상기 버퍼에 프리로드하는 것을 특징으로 하는 재생 장치.

**【청구항 4】**

제1항에 있어서,

상기 버퍼 매니저는 상기 컨텐트 디코더로부터의 fetch 신호에 응답하여 대응하는 마크업 문서를 상기 버퍼에 프리로드하는 것을 특징으로 하는 재생 장치.

**【청구항 5】**

제1항에 있어서,

상기 버퍼 매니저는 상기 컨텐트 디코더로부터의 fetch 신호에 의한 프리로드의 완료 여부를 상기 컨텐트 디코더로 알려주는 것을 특징으로 하는 재생 장치.

**【청구항 6】**

제1항에 있어서,

상기 컨텐트 디코더는 API를 이용하여 상기 fetch 신호를 발생시킴을 특징으로 하는 재생 장치.

**【청구항 7】**

제1항에 있어서,

상기 컨텐트 디코더는 API를 이용하여 상기 버퍼 매니저로 상기 버퍼의 프리로드 완료 여부를 확인하는 것을 특징으로 하는 재생 장치.

**【청구항 8】**

제1항에 있어서,

### 상기 컨텐트 디코더는

대응하는 마크업 문서의 프리로드가 완료되었는지 여부를 API를 사용하여 확인하는 것을 특징으로 하는 재생 장치.

#### 【청구항 9】

제1항에 있어서,

상기 컨텐트 디코더는 상기 프리로드의 완료 여부를 [obj].allDone API를 사용하여 확인하는 것을 특징으로 하는 재생 장치.

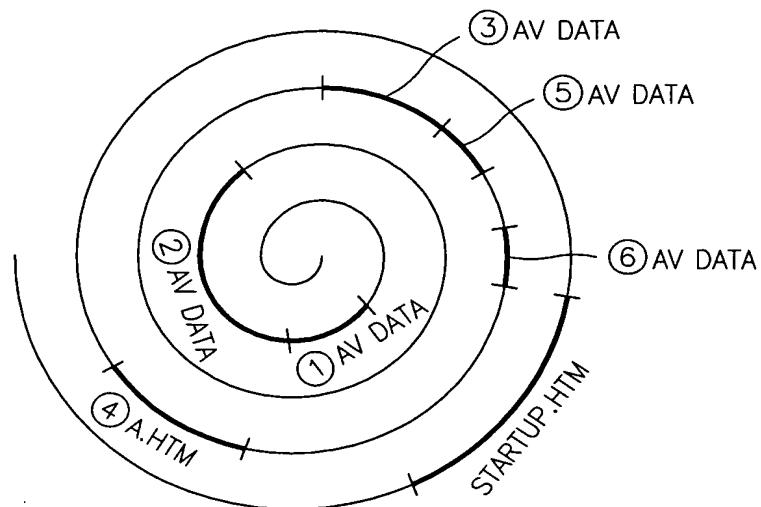
#### 【청구항 10】

제9항에 있어서,

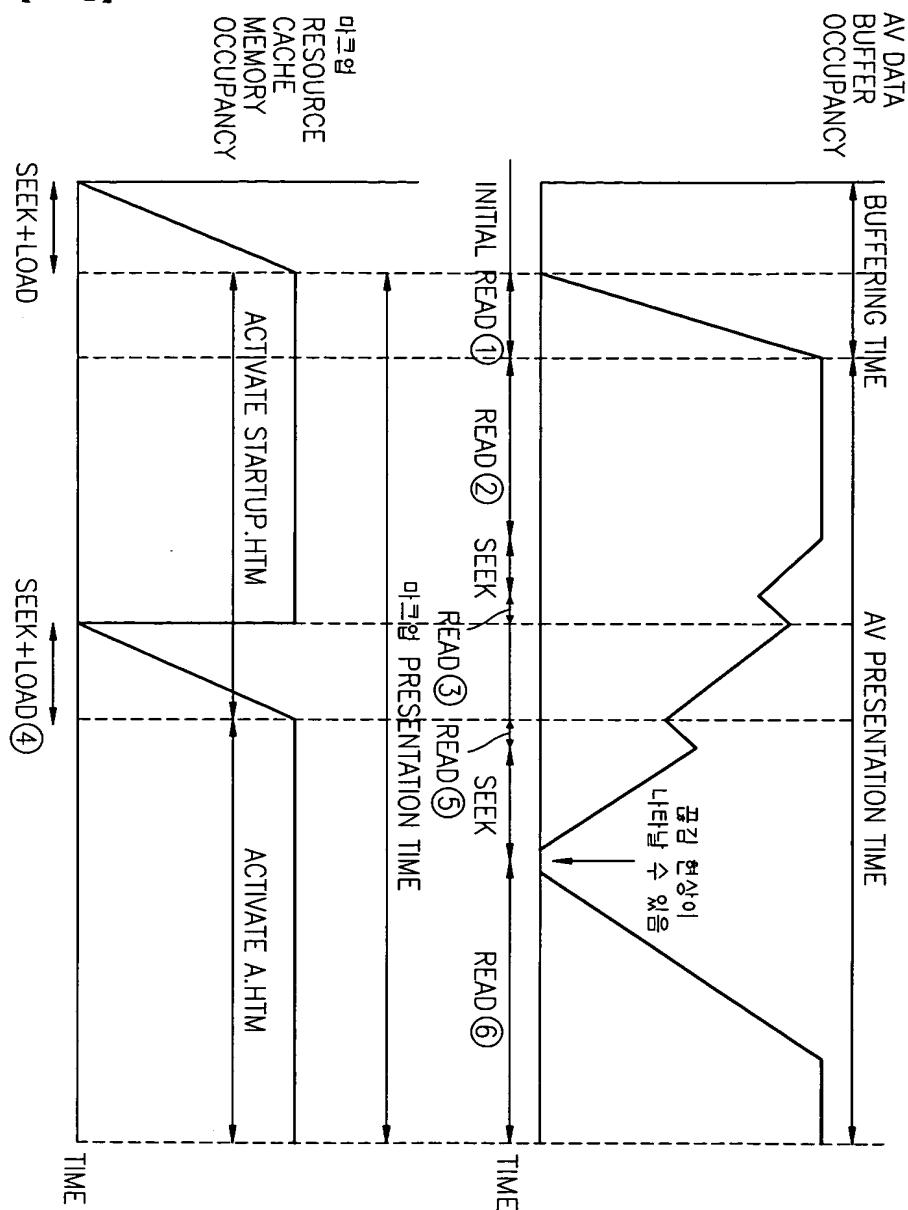
상기 [obj].allDone API는 현재 성공 완료이면 성공을 반환하고, 그렇지 않으면 실패를 반환하는 것을 특징으로 하는 재생 장치.

## 【도면】

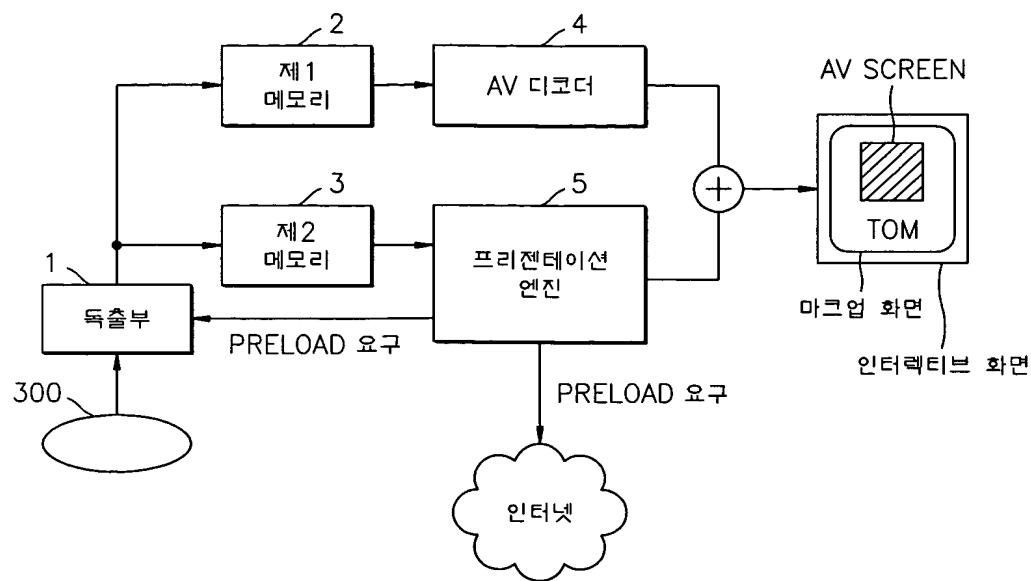
【도 1】



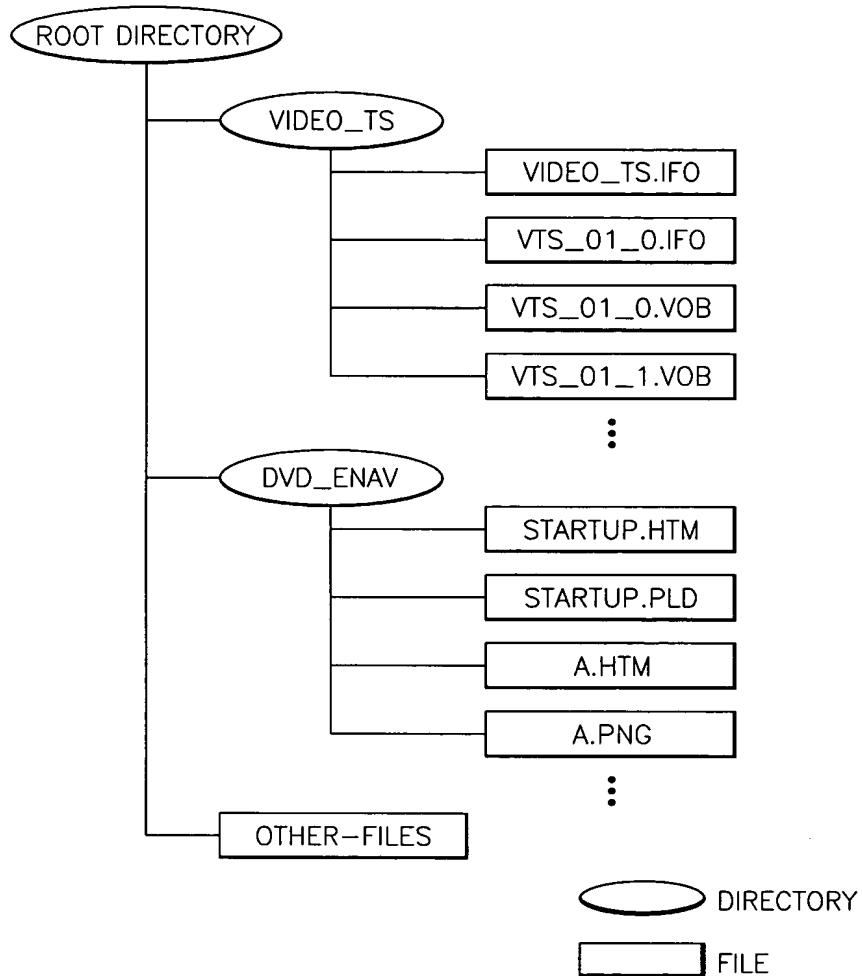
【표 2】



【도 3】



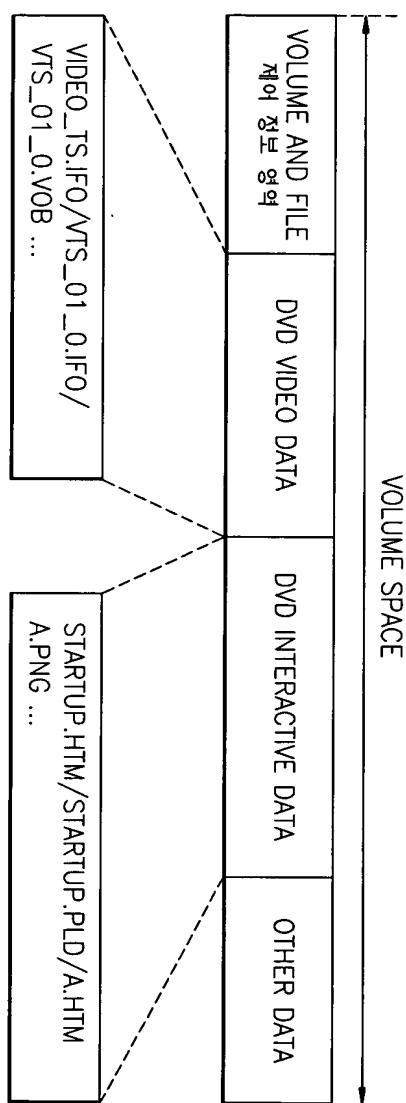
## 【도 4】



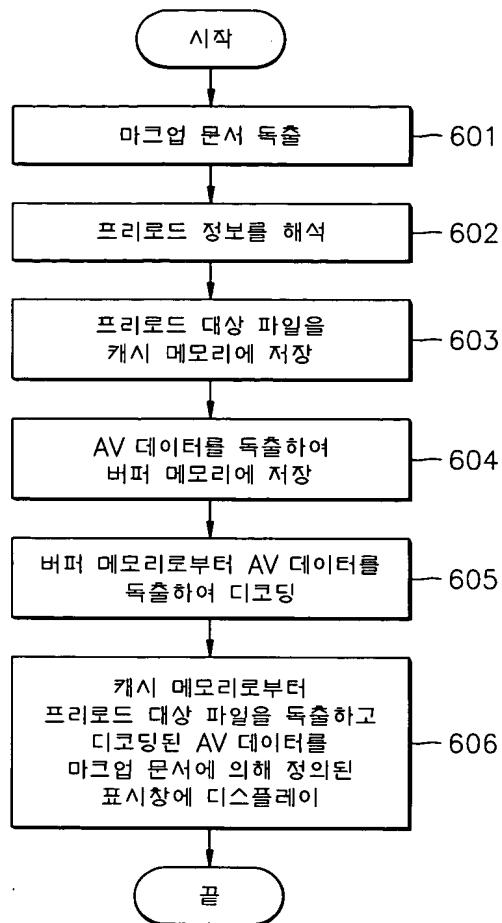
DIRECTORY

FILE

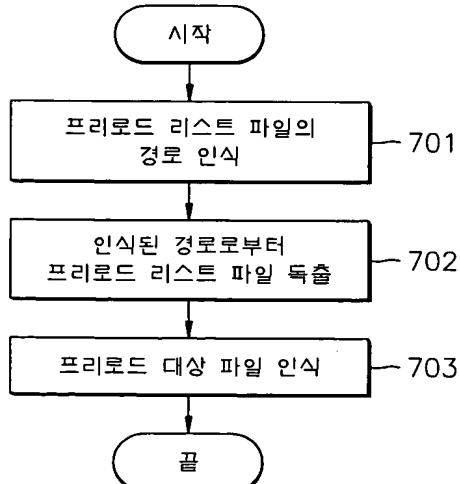
【도 5】



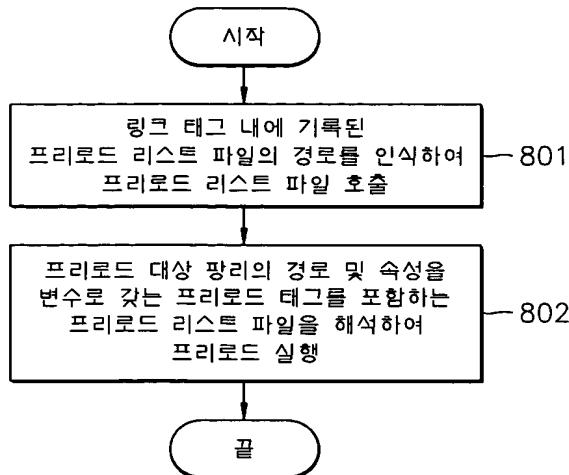
【도 6】



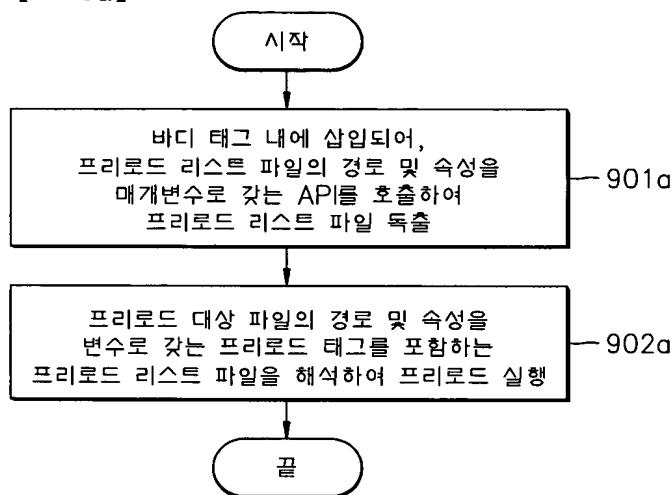
【도 7】



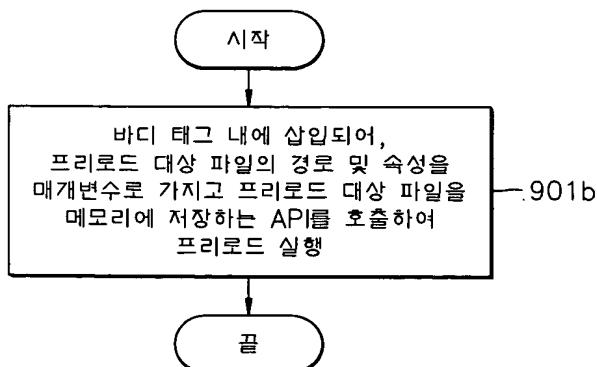
【도 8】



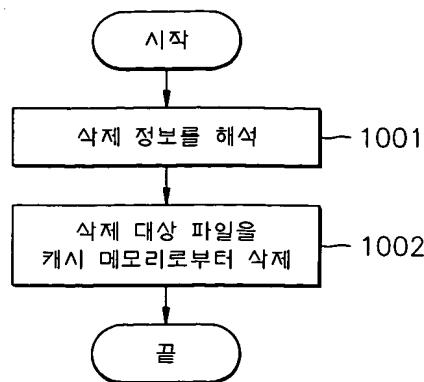
【도 9a】



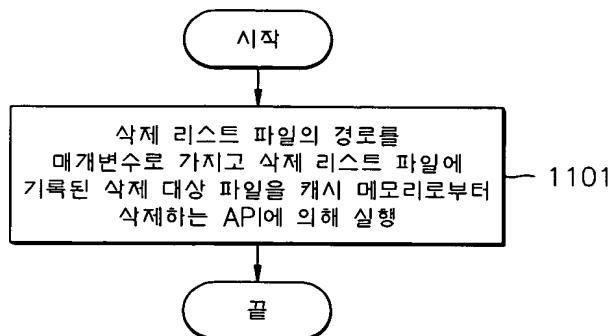
【도 9b】



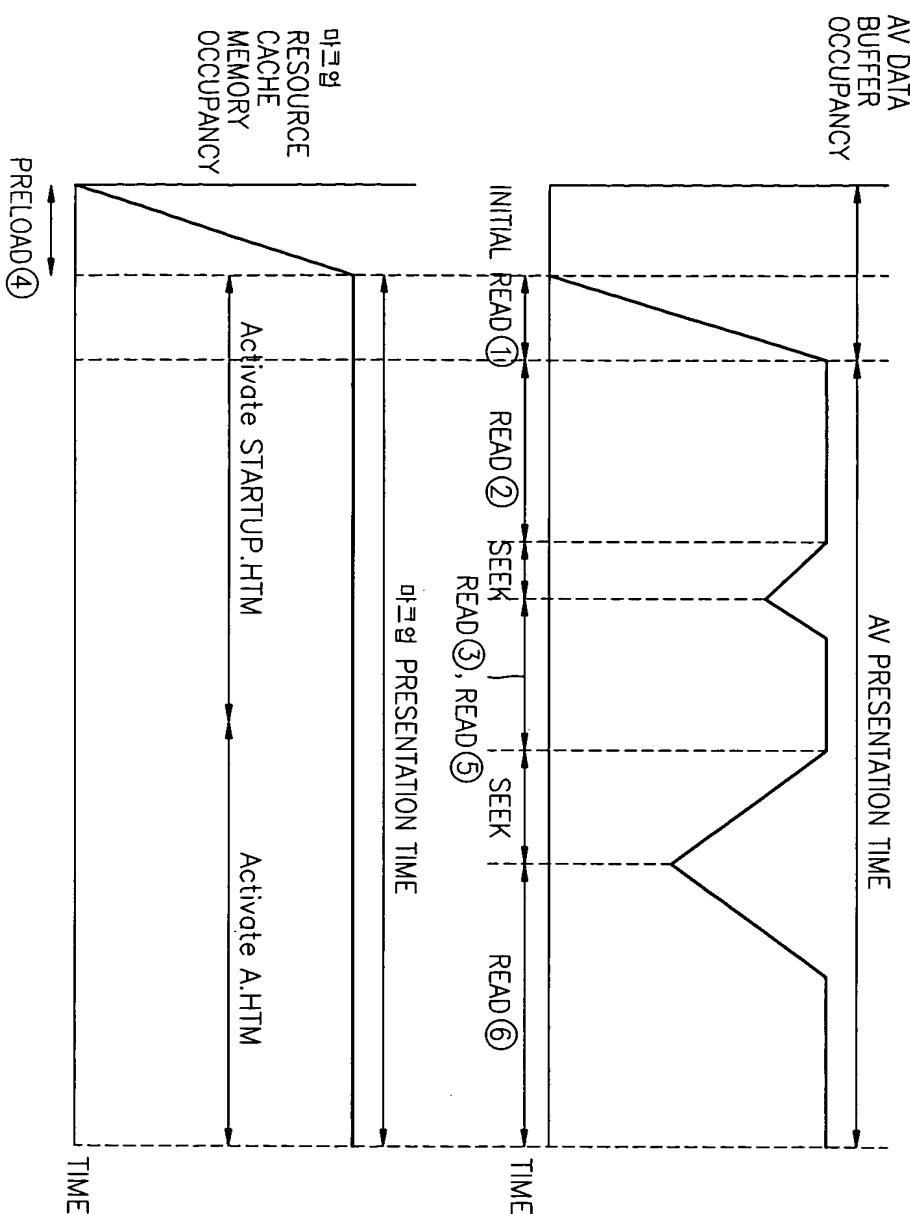
【도 10】



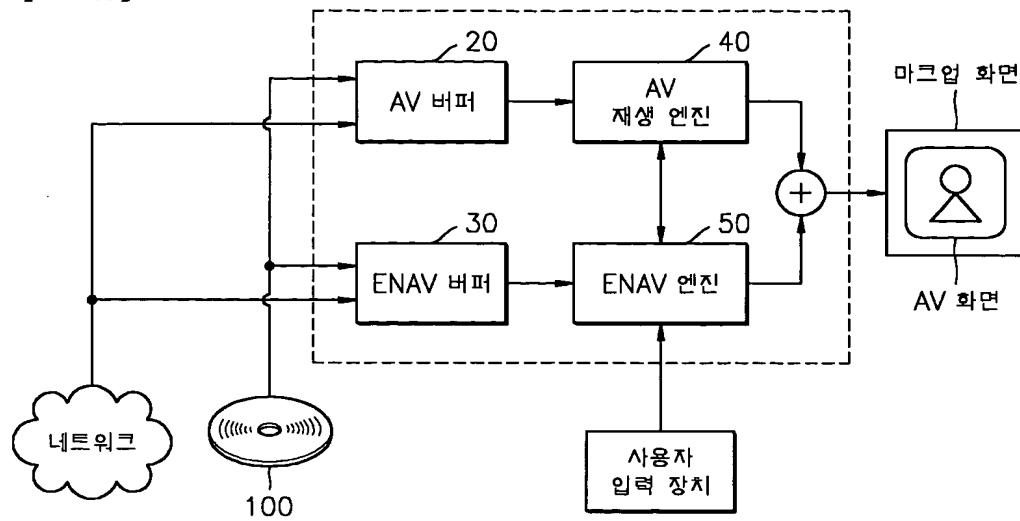
【도 11】



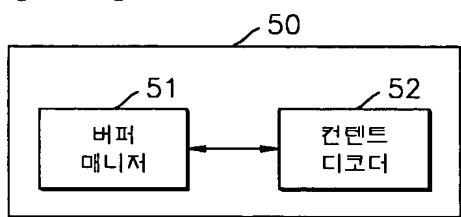
【도 12】



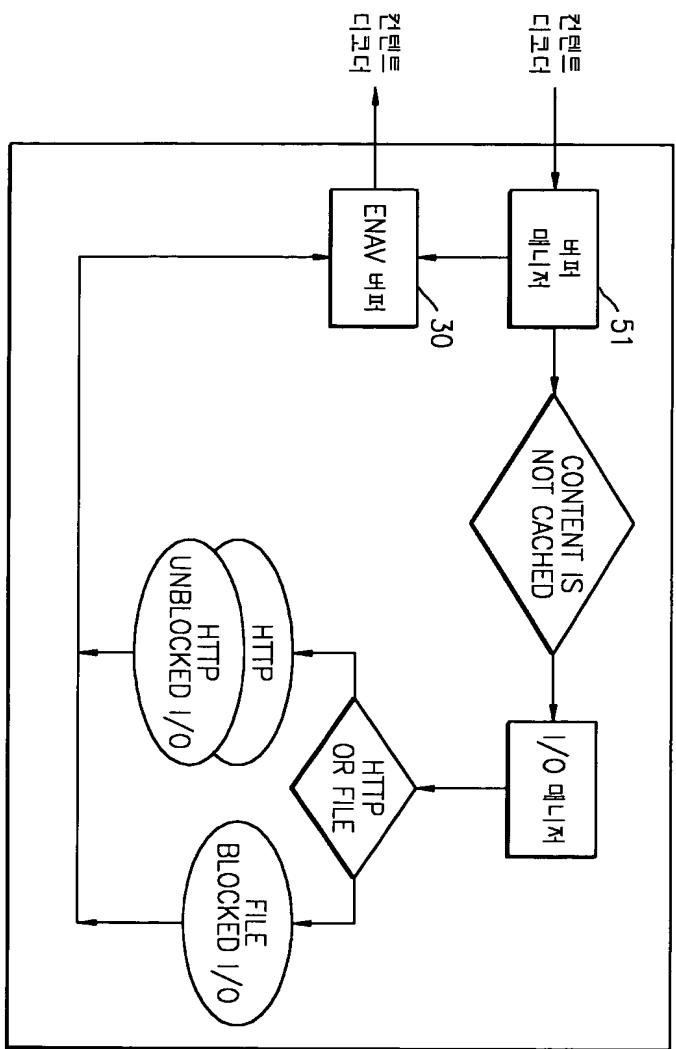
【도 13】



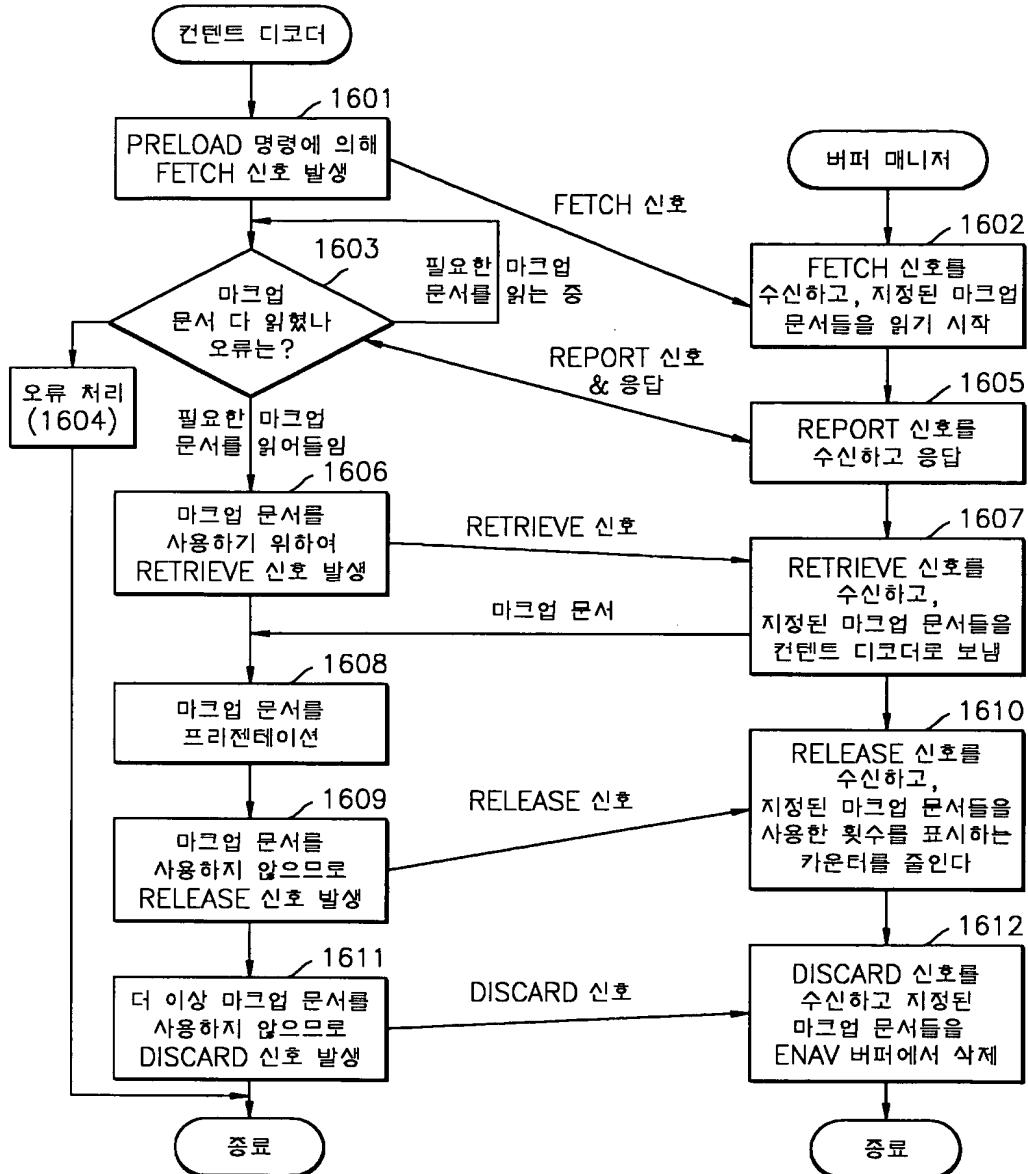
【도 14】



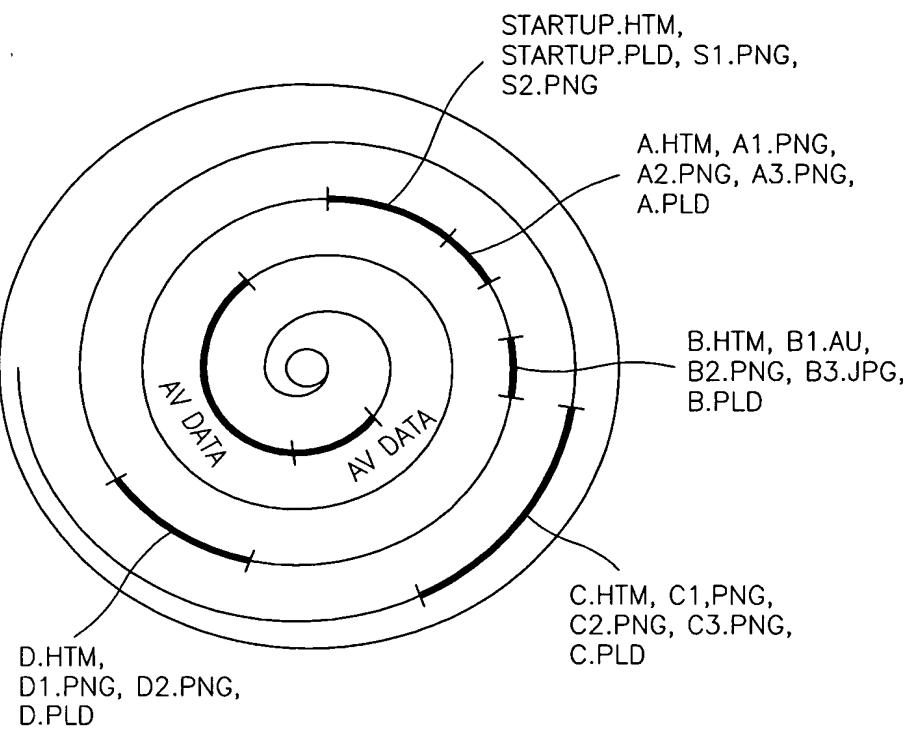
【도 15】



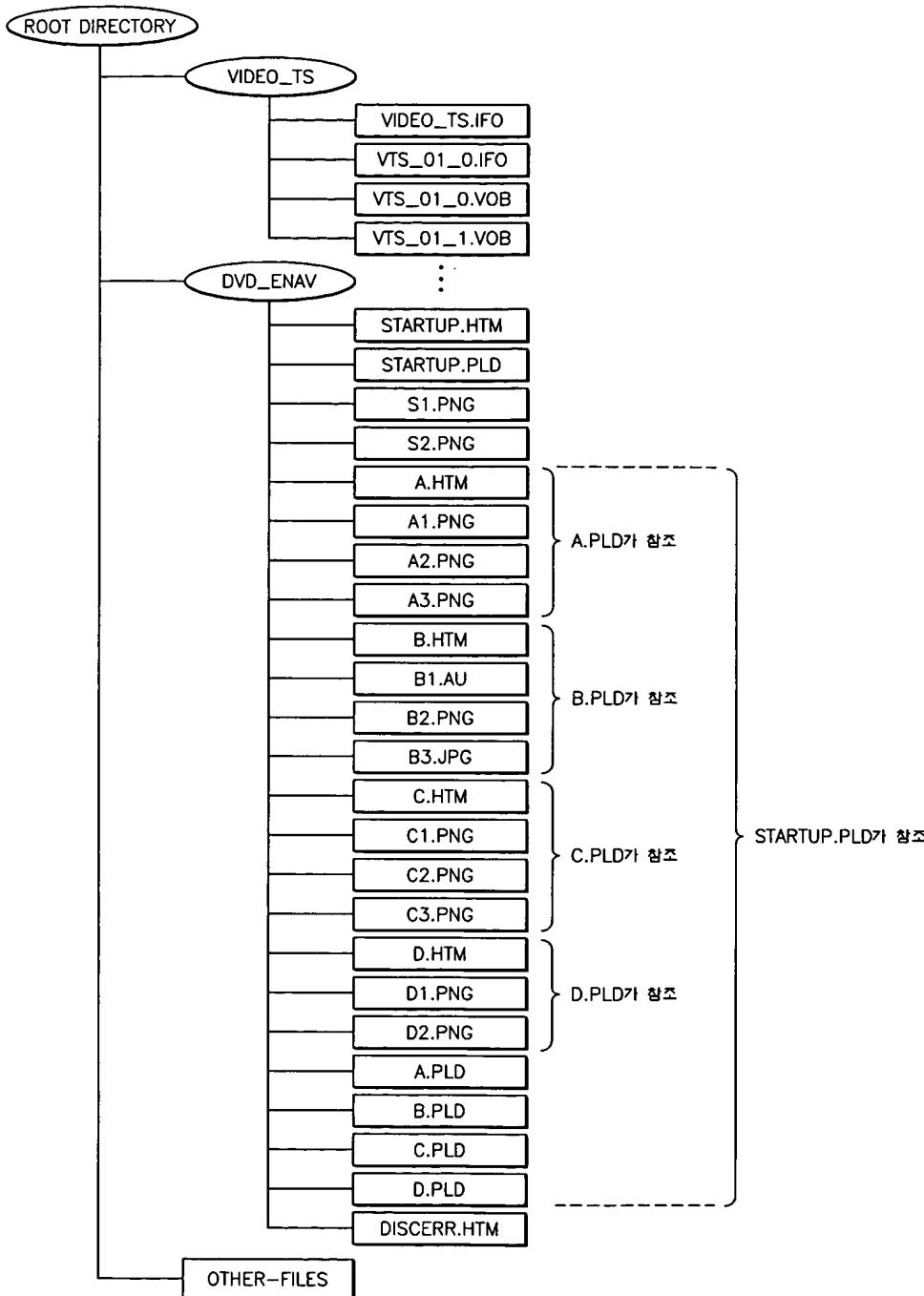
【도 16】



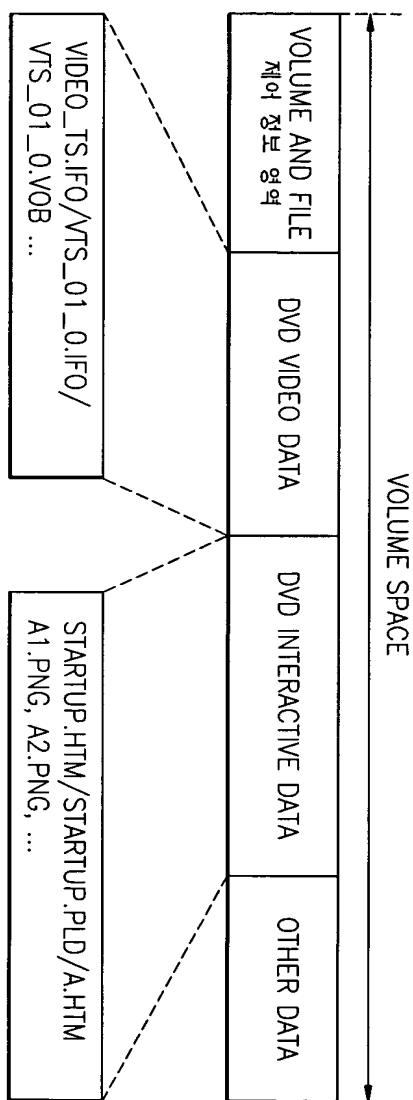
【도 17】



【도 18】



【도 19】



【화 20】

